Python

Orienté objet

Table des matières

Table des matières

Introduction

Paradigme orienté objet

Le paradigme orienté objet a pour objectif de définir des briques logicielles appelées des objets. Ces derniers peuvent représenter des entités physiques ou encore des concepts ou des idées.

Chaque objet possède une structure interne et un comportement.

Le concept d'objet

Il s'agit du point central du paradigme orienté objet.

Un objet est donc une entité modélisant une idée, un concept ou toutes entités physiques du monde réel.

Un objet est caractérisé par :

- Un état
- Un comportement

Coder dans plusieurs fichiers

Pour développer de manière structurée, nous allons coder dans plusieurs fichiers.

Pour que cela soit possible, il sera nécessaire de réaliser un import entre les fichiers.

```
from Package.Fichier import ma_fonction
```

- from → Fichier contenant le code à importer.
  Chemin complet : Répertoire en package + Nom du fichier.
- import → Nom de l'élément à rendre disponible dans le code.
  Il est possible d'importer une fonction, une variable ou une classe.

Exemple d'import

FichierPrincipal.py

```
from FichierExterne import puissance

nombre = 5
exposant = 3

resultat = puissance(nombre,exposant)
print(resultat)
```

FichierExterne.py

```
def puissance(nb, exp):
    resultat = 1
    count = 0
    while count < exp:
        resultat *= nb
        count += 1
    return resultat
```

Limiter l'exécution du code

Lors de l'import d'un fichier, nous ne souhaitons pas exécuter le code de celui-ci.

Il faut donc limiter l'exécution du code au sein du fichier.
Pour cela, il faut utilise le nom du scope accessible à l'aide de la variable "*__name__*".
La valeur de celle-ci vaut "*__main__*" lorsque le script est le point d'entrée.

```
if __name__ == "__main__":
    # execute only if run as a script
    print("Hello World")
```